



Rivest Cipher 4 (RC4)

Dr. Bhasakr Mondal

Stream Cipher

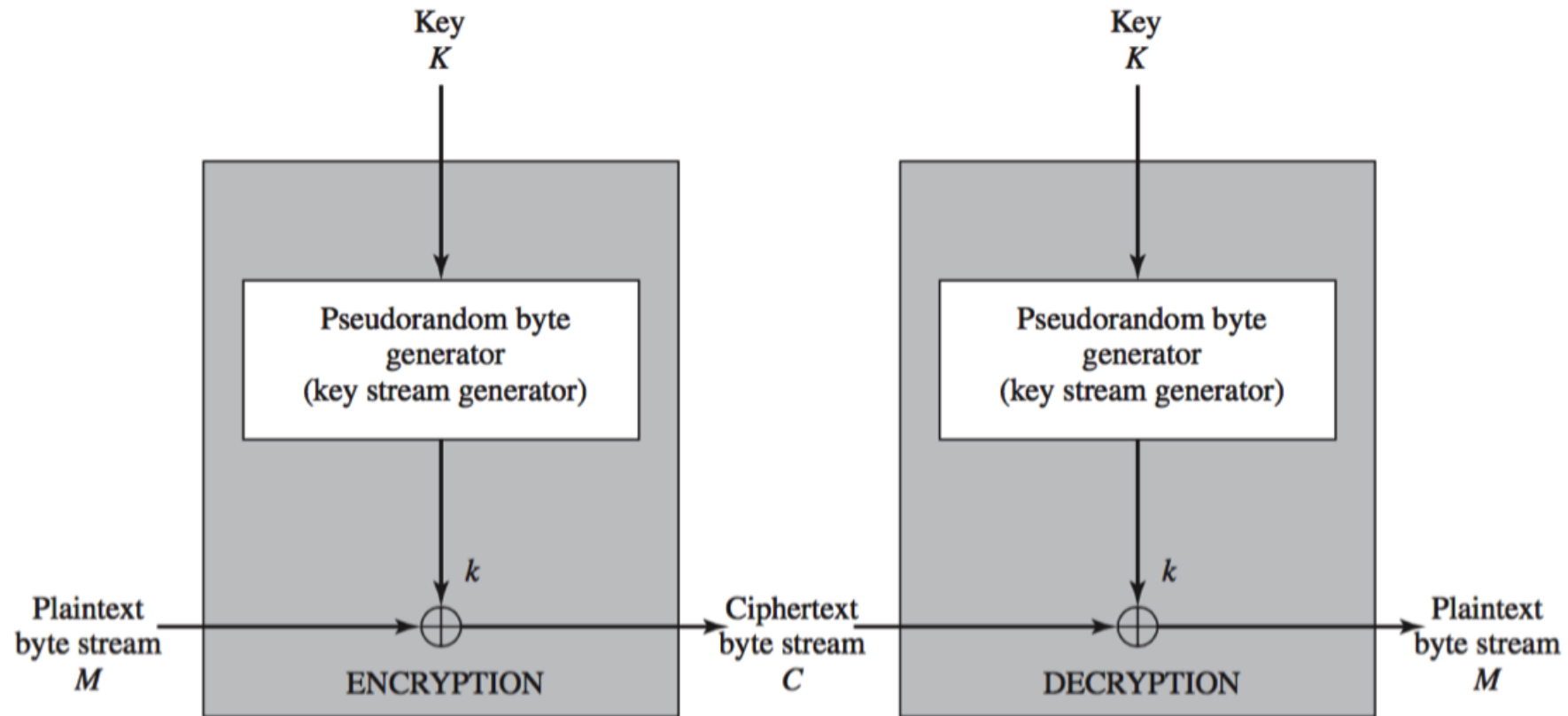


Figure 7.5 Stream Cipher Diagram

Exclusive-OR (XOR) operation

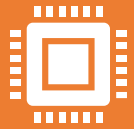
$$\begin{array}{rcl} & 11001100 & \text{plaintext} \\ \oplus & 01101100 & \text{key stream} \\ \hline & 10100000 & \text{ciphertext} \end{array}$$

$$\begin{array}{rcl} & 10100000 & \text{ciphertext} \\ \oplus & 01101100 & \text{key stream} \\ \hline & 11001100 & \text{plaintext} \end{array}$$



RC4 Overview

- RC4 (Ron's Code) was designed by Ron Rivest of RSA Security in 1987
- Rivest Cipher 4 also known as ARC4
- a stream cipher
- variable length key algorithm
- encrypts one byte at a time (or larger units on a time)
- simplicity and speed in software



The algorithm uses a variable length key from 1 to 256 bytes to initialize a 256-byte state table.



The state table is used for subsequent generation of pseudo-random bytes and then to generate a pseudo-random stream which is XORed with the plaintext to give the ciphertext.



Each element in the state table is swapped at least once.

2 Phases



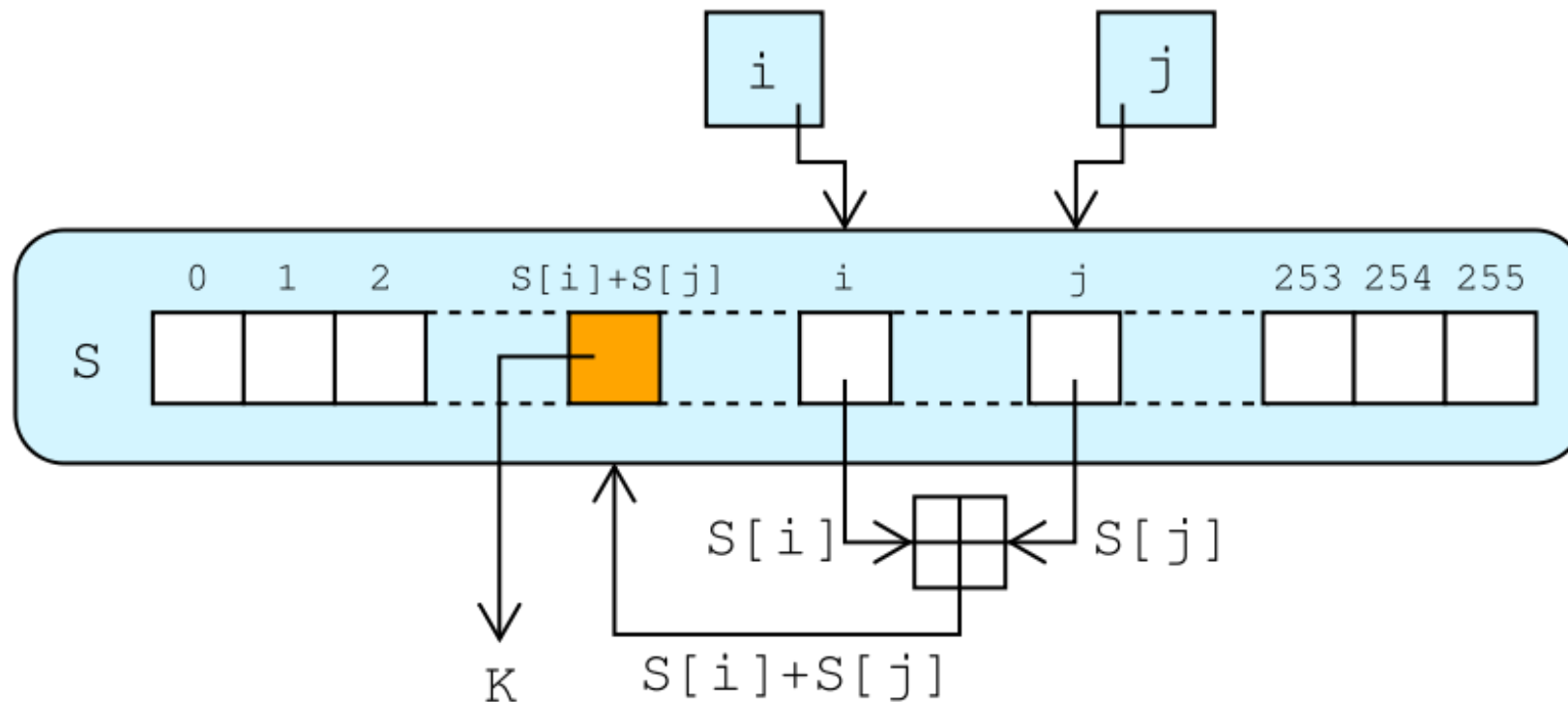
- key setup
 - During a N-bit key setup (N being your key length), the encryption key is used to generate an encrypting variable using two arrays, state and key, and N-number of mixing operations (swapping bytes, modulo operations, and other formulas).
- ciphering

Initialization

- Entries of S are set equal to the values from 0 through 255 in ascending order; that is; $S[0] = 0, S[1] = 1, \dots, S[255] = 255$.
- A temporary vector T is created
- If the length of the key K is 256 bytes, then K is transferred to T
- Otherwise, for a key of length *keylen* bytes, the first *keylen* elements of T are copied from K and then K is repeated as many times as necessary to fill out T .
 - for $i = 0$ to 255 do
 - $S[i] = i$;
 - $T[i] = K[i \bmod \text{keylen}]$;
- use T to produce the initial permutation of S


Initial Permutation of S

- $j = 0;$
- for $i = 0$ to 255 do
 - $j = (j + S[i] + T[i]) \bmod 256;$
 - Swap ($S[i], S[j]$);



Stream Generation

- $i, j = 0;$
- while (true)
 - $i = (i + 1) \bmod 256;$
 - $j = (j + S[i]) \bmod 256;$
 - Swap ($S[i], S[j]$);
 - $t = (S[i] + S[j]) \bmod 256;$
 - $k = S[t];$



$$\mathit{ciphertext}[l] = \mathit{plaintext}[l] \oplus K[l]$$

keys



OFTEN LIMITED TO 40 BITS,
BECAUSE OF EXPORT
RESTRICTIONS



SOMETIMES USED AS A 128 BIT
KEY



HAS THE CAPABILITY OF USING
KEYS BETWEEN 1 AND 2048 BITS.

Strengths of RC4



The difficulty of knowing where any value is in the table.



The difficulty of knowing which location in the table is used to select each value in the sequence.



Encryption is about 10 times faster than DES.

RC4: multiple vulnerabilities



RC4 is no longer considered secure.



One in every 256 keys can be a weak key. These keys are identified by cryptanalysis that is able to find circumstances under which one of more generated bytes are strongly correlated with a few bytes of the key.



especially vulnerable when the beginning of the output keystream is not discarded



vulnerable when nonrandom or related keys are used.

RC4

Example

-
- Assume we use a 4 x 3-bit key, K, and plaintext P as below:
 - $K = [1\ 2\ 3\ 6]$
 - $P = [1\ 2\ 2\ 2]$
 - Initialize the state vector S and temporary vector T. S is initialized so the $S[i] = i$, and T is initialized, so it is the key K (repeated as necessary).
 - $S = [0\ 1\ 2\ 3\ 4\ 5\ 6\ 7]$
 - $T = [1\ 2\ 3\ 6\ 1\ 2\ 3\ 6]$
 - Now perform the initial permutation on S.
 - For $i = 0$:
 - $j = (0 + 0 + 1) \bmod 8 = 1$
 - $\text{Swap}(S[0], S[1]);$
 - So, in the 1st iteration $S[0]$ must be swapped with $S[1]$ giving:
 - $S = [1\ 0\ 2\ 3\ 4\ 5\ 6\ 7]$

$S = [1\ 0\ 2\ 3\ 4\ 5\ 6\ 7]$

$T = [1\ 2\ 3\ 6\ 1\ 2\ 3\ 6]$

$i, j = (j + S[i] + T[i]) \bmod 8$	
$i = 0, j = 1$	$S = [1\ 0\ 2\ 3\ 4\ 5\ 6\ 7]$
$i = 1, j = 3$	$S = [1\ 3\ 2\ 0\ 4\ 5\ 6\ 7];$
$i = 2, j = 0$	$S = [2\ 3\ 1\ 0\ 4\ 5\ 6\ 7];$
$i = 3, j = 6;$	$S = [2\ 3\ 1\ 6\ 4\ 5\ 0\ 7];$
$i = 4, j = 3$	$S = [2\ 3\ 1\ 4\ 6\ 5\ 0\ 7];$
$i = 5, j = 2$	$S = [2\ 3\ 5\ 4\ 6\ 1\ 0\ 7];$
$i = 6, j = 5;$	$S = [2\ 3\ 5\ 4\ 6\ 0\ 1\ 7];$
$i = 7, j = 2;$	$S = [2\ 3\ 7\ 4\ 6\ 0\ 1\ 5];$

Hence, our initial permutation of S gives: $S = [2\ 3\ 7\ 4\ 6\ 0\ 1\ 5];$

RC4 Encryption

- encryption continues shuffling array values
- sum of shuffled pair selects "stream key" value from permutation
- XOR $S[t]$ with next byte of message to en/decrypt

$i, j = 0;$

while (true) {

$i = (i + 1) \bmod 8;$

$j = (j + S[i]) \bmod 8;$

Swap ($S[i], S[j]$);

$t = (S[i] + S[j]) \bmod 8;$

$k = S[t];$

}

RC4 Example

- Now we generate 3-bits at a time, k , that we XOR with each 3-bits of plaintext to produce the ciphertext. The 3-bits k is generated by:
- The first iteration:
 - $S = [2\ 3\ 7\ 4\ 6\ 0\ 1\ 5]$
 - $i = (0 + 1) \bmod 8 = 1$
 - $j = (0 + S[1]) \bmod 8 = 3$
 - $\text{Swap}(S[1], S[3])$
 - $S = [2\ 4\ 7\ 3\ 6\ 0\ 1\ 5]$
 - $t = (S[1] + S[3]) \bmod 8 = 7$
 - $k = S[7] = 5$
- So our first 3-bits of ciphertext is obtained by:
- $k \text{ XOR } P(1) = 5 \text{ XOR } 1 = 101 \text{ XOR } 001 = 100 = 4$

RC4 Example

iteration=1, k=5	$5 \text{ XOR } 1 = 101 \text{ XOR } 001 = 100 = 4$
iteration=2, k=1	$1 \text{ XOR } 2 = 001 \text{ XOR } 010 = 011 = 3$
iteration=3, k=0	$0 \text{ XOR } 2 = 000 \text{ XOR } 010 = 010 = 2$
iteration=3, k=1	$1 \text{ XOR } 2 = 001 \text{ XOR } 010 = 011 = 3$

$P = [1 \ 2 \ 2 \ 2]$

$K = [5 \ 1 \ 0 \ 1]$

$C = [4 \ 3 \ 2 \ 3]$

RC4 Security

- claimed secure against known attacks
- However, have some analyses, some on the verge of practical
- first 256 bytes have bias
- multiple keys/same plaintext attack
- result is very non-linear
- since RC4 is a stream cipher, must never reuse a key
- have a concern with WEP, but due to key handling rather than RC4 itself

References

- William Stallings, Network Security Essentials : Applications and Standards, ISBN: 9788131761755, 8131761754
- Thanks to the many unknown sources from where some information is adopted.



Thank You

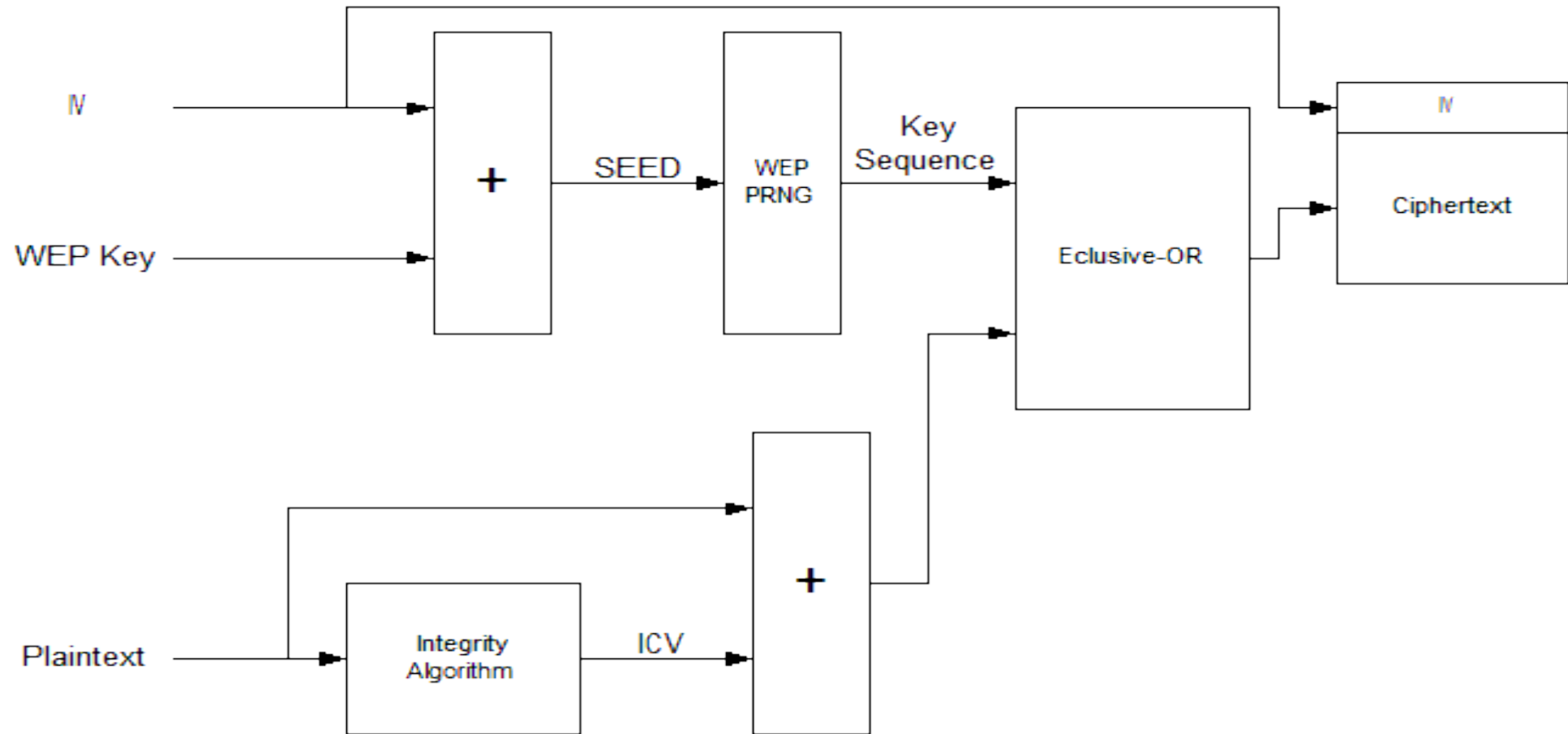


Figure 2: WEP Encryption

Wired Equivalent Privacy (WEP) or Rivest Cipher 4 (RC4)
Encryption

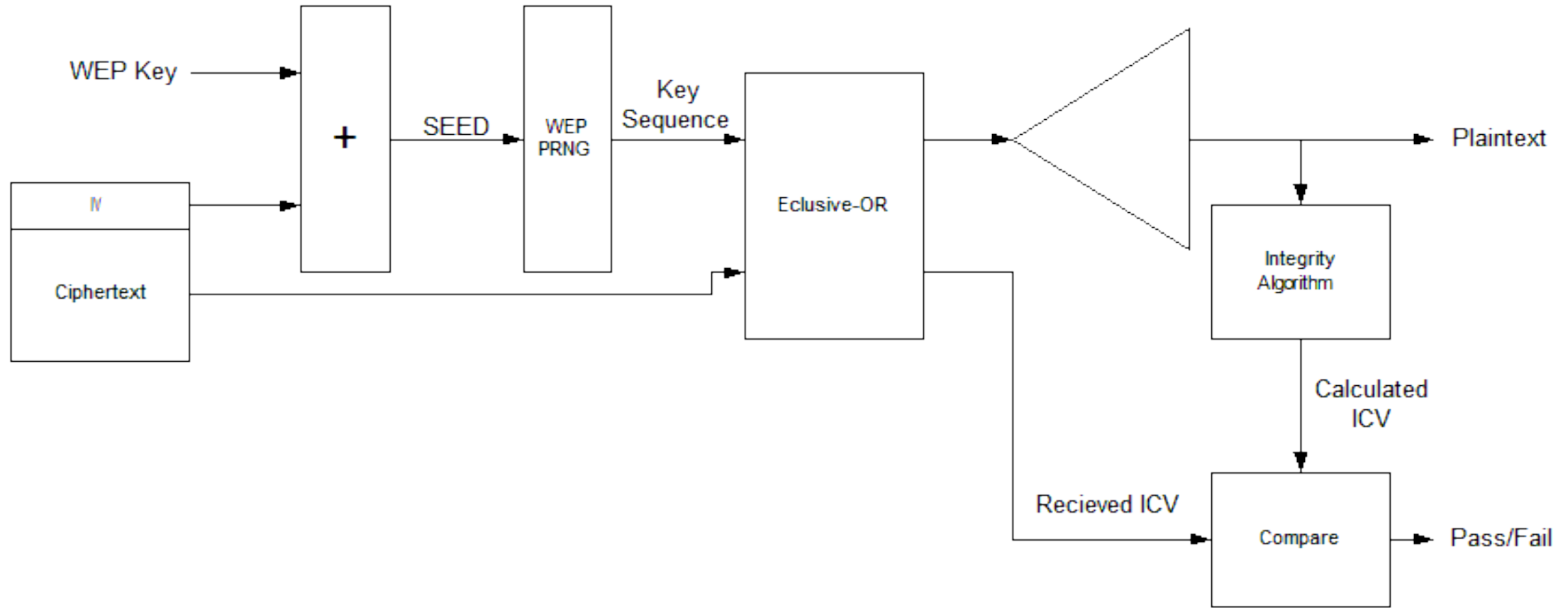


Figure 3: WEP Decryption

WEP or Rivest Cipher 4 (RC4)
Decryption

Wi-Fi Protected Access 2 (WPA 2)

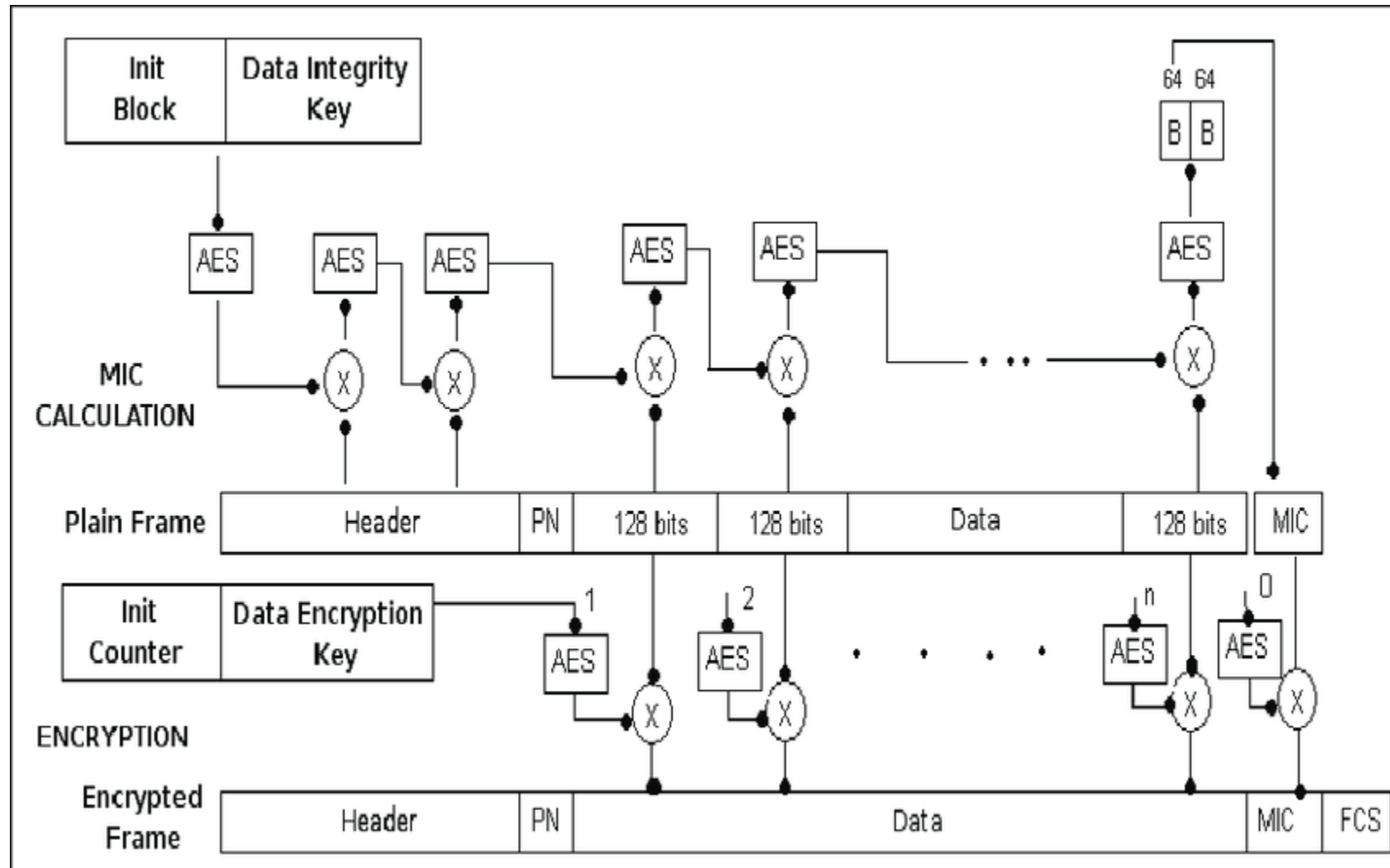


Table 1. Main features of WEP, WPA, and WPA-2

	<i>WEP</i>	<i>WPA</i>	<i>WPA-2</i>
Authentication	N/A	IEEE 802.1X/ EAP/PSK	IEEE 802.1X/ EAP/PSK
Cryptographic algorithm	RC4	RC4	AES
Key size	40 O 104 bits	128 bits	128 bits
Encryption method	WEP	TKIP	CCMP
Data integrity	CRC32	MIC	CCM
Keys for packets	No	Yes	Yes
IV length	24 bits	48 bits	48 bits